

Storage Engine, Provisioning Assurance Request Failed

Ericsson Centralized User Database

OPERATING INSTRUCTION

Copyright

© Ericsson AB 2016, 2017. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 1 |
| 1.1 | Alarm Description | 1 |
| 1.2 | Prerequisites | 3 |
| 2 | Procedure | 5 |
| 2.1 | Actions for Incorrectly Defined PG Endpoints | 5 |
| 2.2 | Actions for Network Connection Problems | 5 |
| 2.3 | Actions for HTTP Authentication Error | 14 |
| 2.4 | Common Actions after Alarm Causes are Fixed | 14 |
| | Glossary | 15 |
| | Reference List | 17 |



Storage Engine, Provisioning Assurance Request Failed



1 Introduction

This instruction concerns alarm handling for the Storage Engine, Provisioning Assurance Request Failed alarm.

1.1 Alarm Description

This alarm is raised if the Ericsson Centralized User Database (CUDB) site tries to contact the Provisioning Gateway (PG) to initialize or request the status of the Provisioning Assurance execution, but the PG cannot be reached because of an HTTP communication timeout or an authorization error.

The alarm is issued in the following situations:

- Incorrectly defined PG endpoints.
- Network connection problems.
- HTTP authentication error.

The possible alarm causes and the corresponding fault reasons, fault locations, and impacts are described in Table 1.

Table 1 Alarm Causes

| Alarm Cause | Description | Fault Reason | Fault Location | Impact |
|-----------------------------------|--|---|---|---|
| Incorrectly defined PG endpoints. | Provisioning Gateway in URL <Request/Reply URL> cannot be reached. | PG endpoint definitions are configured incorrectly. | CUDB data model. | Provisioning assurance not started. |
| Network connection problems. | A temporary connectivity issue causes replay failure. | Communication error. | <ul style="list-style-type: none"> Provisioning system. eVIP level Infrastructure level: BSP CMX⁽¹⁾ or cloud infrastructure⁽²⁾ Outside the CUDB system. | Replay failure in provisioning assurance. |
| HTTP authentication error. | Authentication failure when starting provisioning assurance. | Faulty configuration of HTTP authentication. | CUDB data model. | Provisioning assurance not executed. |

(1) For CUDB systems deployed on native BSP 8100.

(2) For CUDB systems deployed on a cloud infrastructure.

Note: An alarm can appear as a result of maintenance activity.

The following are the consequences for the node if the alarm is not solved:



Non-smooth mastership change can cause possible provisioning data loss. This can happen during provisioning, where it was not possible to replicate all of the provisioning operations to the slaves.

The alarm attributes are listed and explained in Table 2.

Table 2 Alarm Attributes

| Attribute Name | Attribute Value |
|------------------------------|--|
| Auto Cease | No |
| Module | STORAGE - ENGINE |
| Error Code | 23 |
| Timestamp First | Date and time when the alarm was raised for the first time. |
| Repeated Counter | Number which indicates how many times the alarm was raised. |
| Timestamp Last | Date and time of the most recent alarm raised. |
| Resource ID | .1.3.6.1.4.1.193.169.1.3.2.<PG endpoint ID>.<Reprovisioning Timestamp> |
| Alarm Model Description | Provisioning Assurance Request Failed, Storage Engine |
| Alarm Active Description | Storage Engine: Provisioning Assurance request with id <Reprovisioning Timestamp> to PG failed. Provisioning Gateway in URL <Request/Replay URL> cannot be reached due to <Error Description>. |
| ITU Alarm Event Type | communicationsAlarm (2) |
| ITU Alarm Probable Cause | communicationSubsystemFailure (505) |
| ITU Alarm Perceived Severity | (4) - Major |
| Originating Source IP | Node ID where the alarm was raised. |
| Sequence Number | Number which indicates the order in which alarms were raised. |

In Table 2, the indicated variables are as follows:

- <PG endpoint ID> is the PG identifier.
- <Request/Replay URL> is the Request URL or Replay URL of the PG.
- <Reprovisioning Timestamp> is the point in time from which CUDB may have missing provisioning operations, expressed as POSIX time in seconds.
- <Error Description> is the root cause of the communication error, in case it can be identified. The possible values are as follows:
 - communication error
 - authentication failure
 - unknown http answer
 - BUSY answer code from PG more than <time_waiting> seconds



In the above error message `<time_waiting>` is the time in seconds spent by CUDB waiting for an OK answer from the PG before starting a new provisioning assurance.

For more information about the attribute descriptions, refer to [CUDB Node Fault Management Configuration Guide, Reference \[1\]](#).

1.2 Prerequisites

This section provides information on the documents, tools, and conditions that apply to the procedure.

1.2.1 Documents

Before starting this procedure, ensure that you have read the following documents:

- [CUDB Node Fault Management Configuration Guide, Reference \[1\]](#), regarding alarm configuration.
- [CUDB System Administrator Guide, Reference \[2\]](#), regarding the steps of configuring PG endpoints for Provisioning Assurance.
- [System Safety Information, Reference \[5\]](#).
- [Personal Health and Safety Information, Reference \[6\]](#).

1.2.2 Tools

Not applicable.

1.2.3 Conditions

This alarm can only be raised when Provisioning Assurance is active.



Storage Engine, Provisioning Assurance Request Failed



2 Procedure

This section describes the procedure to follow when this alarm is received.

2.1 Actions for Incorrectly Defined PG Endpoints

Do the following:

1. Check the definition of the PG endpoints (refer to [CUDB Node Configuration Data Model Description](#), Reference [3] for more information about the configuration of PG endpoints). In case the configuration is incorrect, configure it again properly by following the steps of [CUDB System Administrator Guide](#), Reference [2].
2. Check if the CUDB system and the PG are properly connected. To do so, execute the following command from any CUDB System Controller (SC):

```
SC_2_<x># telnet <pg_ip> <pg_port>
```

In the above command, <pg_ip> stands for the PG IP address, while <pg_port> is the port number of the HTTP service in the PG.

When CUDB-PG connectivity is OK, you receive the following printout: Connected to <pg_ip>. In this case, finish the telnet session by entering Ctrl-C and continue troubleshooting with the next step.

When CUDB-PG connectivity is failing, you receive the following printout: Connection refused. In this case, before performing additional troubleshooting in CUDB, verify that the HTTP service and port in PG are working properly.

3. If none of the above preliminary checks show any errors, then the automatic replay failure is caused by a temporary connectivity issue. In this case, follow the steps of Section 2.2 on page 5.

2.2 Actions for Network Connection Problems

Do the following:

1. If the check of Step 2 indicates a connection error, then check that the provisioning system is correctly up and running. If it is not, then fix the problem at the provisioning system and repeat the check described in Step 1. The specific actions to perform on the provisioning system are out of the scope of this Operating Instruction. If the connectivity between the CUDB system and the PG is restored, continue the procedure as described in Section 2.3 on page 14; if it is not, continue with the next step.



2. If the connection problem persists, execute a `traceroute` from any of the SCs to identify the specific point where connectivity is failing. Use the following command to do so:

```
SC_2_<x># traceroute -n <PG_OAM_VIP>
```

For CUDB systems deployed on native BSP 8100, the expected output must be similar to the below example :

```
traceroute to <PG_OAM_VIP>(<PG_OAM_VIP>) ...
1 * * *
2 <CUDB_OAM_VIP> (<CUDB_OAM_VIP>) <xxx> ms ...
3 <FEE_ALB0_IP> (<FEE_ALB0_IP>) <xxx> ms ...
4 <BSP CMX IP> (<BSP CMX IP>) <xxx> ms ...
5 <BACKBONE ROUTER IP> (<BACKBONE ROUTER IP>) <xxx> ms ...
```

This process can result in three ways. The connectivity problem is either on:

- eVIP level,
- Infrastructure level: BSP CMX or cloud infrastructure level, or
- outside the CUDB system.

Depending on the results, perform the steps included in Section 2.2.1 on page 6, or Section 2.2.2.1 on page 9, or Section 2.2.3 on page 13, respectively.

2.2.1 Connectivity Problem on eVIP Level

If `traceroute` stops at hop 1 or 2, then the connectivity problem is located on eVIP level. Do the following:

1. Check `syslog`.

To start troubleshooting, check `syslog` with the following command: `less /var/log/messages`

Look for errors that indicate problems with eVIP or starting of eVIP. This can be any error message related to an eVIP process, coming from eVIP itself or other system components. Most obviously, the message contains string "evip" and is of high priority.

2. Check VIP address.

eVIP is a networking function that is implemented almost only with existing standard Linux components. This means that it is recommended to check eVIP with standard Linux networking tools.

eVIP makes a Virtual IP (VIP) address visible to the outside world. The traffic is distributed to individual processors in a cluster. The VIP address is assigned to a device on all nodes in the target group.



Check that the VIP address is assigned to a (preselection) device with `ifconfig` or `ip` on the relevant nodes, see Example 1.

```
> ifconfig
...
alb1      Link encap:UNSPEC  HWaddr FE-80-00-00-00-00-00-00-00-...
          inet addr:10.0.0.1  P-t-P:10.0.0.1  Mask:255.255.255.255
          inet6 addr: 2011::11/128 Scope:Global
          inet6 addr: fe80::200:ff:fe01:1/64 Scope:Link
          inet6 addr: 2011::1/128 Scope:Global
          UP POINTOPOINT RUNNING NOARP  MTU:1452  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Example 1 Check of VIP Address

3. Check Command Line Interface.

eVIP publishes a Command Line Interface (CLI) on the System Controllers (SCs), where `evipc` executes. Only the CLI on the active SC can be used. The address to the active SC is found by using the `getactivecontrol` utility. The CLI is reached with `telnet` on port 25190.

To troubleshoot, perform the following steps:

- a. Login to the eVIP CLI, see Example 2.
- b. Check the eVIP status with the following command: **show evip-status**. STATUS OK does not always indicate that everything is OK.
- c. Check the agents with the following command: **show agents**.

```
/ # /opt/vip/bin/getactivecontrol
fe80::1234%evip_macvlan0
/ # telnet `/opt/vip/bin/getactivecontrol` 25190
```

```
Entering character mode
Escape character is '^]'.
```

```
...
EVIP> show agents
+-----[ ALB alb1 (ACTIVE) ]-----+
+----- PN -----+
| pagent (6)                | lbesl_pn (10)                |
|[4] fe80::ff:fe01:12 : ACTIVE | [4] fe80::ff:fe01:12 : ACTIVE |
|[3] fe80::ff:fe01:f : ACTIVE  | [3] fe80::ff:fe01:f : ACTIVE  |
|[2] fe80::ff:fe01:c : ACTIVE  | [2] fe80::ff:fe01:c : ACTIVE  |
|[1] fe80::ff:fe01:9 : ACTIVE  | [1] fe80::ff:fe01:9 : ACTIVE  |
+-----+-----+
| ersipc (0)                | repdb (10)                |
```



```

|[4] fe80::ff:fe01:12 : ACTIVE      |[4] fe80::ff:fe01:12 : ACTIVE      |
|[3] fe80::ff:fe01:f : ACTIVE      |[3] fe80::ff:fe01:f : ACTIVE      |
|[2] fe80::ff:fe01:c : ACTIVE      |[2] fe80::ff:fe01:c : ACTIVE      |
|[1] fe80::ff:fe01:9 : ACTIVE      |[1] fe80::ff:fe01:9 : ACTIVE      |
+----- LBE -----+
| lbeagent (32)                    | lberrep (8)                      |
|[4] fe80::ff:fe01:4 : ACTIVE      |[4] fe80::ff:fe01:4 : ACTIVE      |
|[3] fe80::ff:fe01:3 : ACTIVE      |[3] fe80::ff:fe01:3 : ACTIVE      |
+-----+
| sesel_lbe (8)                    |                                  |
|[4] fe80::ff:fe01:4 : ACTIVE      |                                  |
|[3] fe80::ff:fe01:3 : ACTIVE      |                                  |
+----- FE -----+
| feeagent (12)                    | lbesel_fe (12)                   |
|[2] fe80::ff:fe01:7 : ACTIVE UP  |[2] fe80::ff:fe01:7 : ACTIVE      |
+-----+
| sesel_fe (8)                     |                                  |
|[2] fe80::ff:fe01:7 : ACTIVE      |                                  |
+----- SE -----+
| seagent (10)                     | lbesel_se (14)                   |
|[4] fe80::ff:fe01:6 : ACTIVE RDY |[4] fe80::ff:fe01:6 : ACTIVE      |
|[3] fe80::ff:fe01:5 : ACTIVE RDY |[3] fe80::ff:fe01:5 : ACTIVE      |
+-----+
| sesel_se (6)                     |                                  |
|[4] fe80::ff:fe01:6 : ACTIVE      |                                  |
|[3] fe80::ff:fe01:5 : ACTIVE      |                                  |
+----- IPSEC -----+
| ikeagent (0)                     | ipsecuagent (8)                  |
|                                  |[4] fe80::ff:fe01:14 : ACTIVE RDY |
|                                  |[3] fe80::ff:fe01:11 : ACTIVE RDY |
|                                  |[2] fe80::ff:fe01:e : ACTIVE RDY  |
|                                  |[1] fe80::ff:fe01:b : ACTIVE RDY  |
+----- XALBSEL -----+
| xalbsel (6)                       |                                  |
|[4] fe80::ff:fe01:13 : ACTIVE      |                                  |
|[3] fe80::ff:fe01:10 : ACTIVE      |                                  |
|[2] fe80::ff:fe01:d : ACTIVE      |                                  |
|[1] fe80::ff:fe01:a : ACTIVE      |                                  |
+-----+
+-----+
eRSIP state: ACTIVE                cIPSEC state: ACTIVE RDY

```

OK

Example 2 Check of eVIP Status

4. Check processes.

The eVIP base processes must be executing and are as follows:



| | |
|----------------|--|
| evipc | The eVIP control plane. These processes execute in an active/standby configuration on some nodes. The control process is started the SCs (SC-1 and SC-2) on a Core MW cluster. |
| factory | These processes are started on all nodes. The factory process handles eVIP tasks on each node on order from the control plane. |

The control and factory processes are started with scripts on system start up. Check that these processes are executing:

```
> ps
...
266 root      34628 S      /opt/vip/bin/evipc -o /var/log/evipc.log \
-b /opt/vip/etc/evipconf.xsd -f /home/evip/evip.xml -s 3 -l 10 -S 0
339 root      40296 S      /opt/vip/bin/factory -o /var/log/factory.log \
-s 3 -l 10 -S 0 -P /home/evip/evip.swl
```

2.2.2 Connectivity Problem on the Infrastructure Level

2.2.2.1 BSP CMX

For CUDB systems deployed on native BSP 8100, and traceroute stops at hop 3, then the connectivity problem is located at BSP CMX level.

Log in CMX

All of the shells described below are possible to access through ssh or RS232 front interface. There are following security restrictions:

- The front management interface (combined debug connector for RS232 and eth2) is disabled by default at CMX startup.
- Root access is blocked for login. User must login as Advanced first and then make a su login as root.
- Telnet is allowed only for the backplane interfaces.

The board also needs an IP address. The IP addresses are obtained automatically by Dynamic Host Configuration Protocol (DHCP) and could be changed manually using `ifconfig` command. There is no way to login to the board without configured IP address at least for one of the backplane interfaces (eth0, eth1).

To be able to access the board using front management interfaces the ports has to be enabled from backplane Ethernet ports either by:

- Linux CLI command (Advanced user) “`interfacectl`” from the Linux shell. For example:

```
interfacectl rs232 up 0
interfacectl ethernet up 0
```



- SNMP command from the CMX or management station: GEN-SYSMGR-MIB object to enable RS232:

```
snmpset -v 2c -c NETMAN <cmx_ip>  
1.3.6.1.4.1.193.177.2.2.5.3.1.1.3.0 i 1
```

GEN-SYSMGR-MIB object to enable eth2:

```
snmpset -v 2c -c NETMAN <cmx_ip>  
1.3.6.1.4.1.193.177.2.2.5.3.1.1.3.1 i 1
```

On logging in, the Linux shell is available for access to the CMX. The sections below explain each of the shells available for configuration in more detail.

Note: These shells are only for debugging and not for OAM purposes.

Linux Shell

Linux is the basic operating system in use on a CMX. The CMX uses a Wind River Linux distribution. The Linux distribution used for the board has a normal set of Linux commands.

Available users:

- basic
- advanced
- root

It is advised that the “root” login is used sparingly and only when a specific operation is not possible with an “advanced” user access.

ZebOS Shell

ZebOS shell is available for the purposes of L3 configuration.

From Linux shell, login to ZebOS shell using `imish` command.

```
advanced@:/advanced> imish
```

In this shell it is possible to view configurations by using the `show` command. Example of commands: `show interface`, `show arp`, `show ip route`. Typing `?` at the prompt displays the list of commands available at that level.

Command `en` (enable) will put shell in privileged mode with full access to the router.

Command `conf term` (configure terminal) will put the shell in configuration mode. In this mode it is possible to turn on and off traces and to take up and down ports and interfaces. `conf term` mode is terminated by the `exit` command.

To terminate the `imish` mode, use the `exit` command.

ZebOS CLI does not support `traceroute` and `telnet` commands.



Tip: When analyzing counters it can be useful to clear the counters first. This can be done using the objects from IF-MIB.

BCM Shell

The Broadcom shell is accessed from a Linux shell that has “root” or “advanced” users logged on. To access the bcm shell, type the following command.

```
root@:/root> bcm
```

The BCM shell provides access to configure and display settings of the switch. Useful commands: `show counters`, `vlan show`, `l2 show`, `ports`, `help` (to display brief description about all commands), `exit` (to terminate BCM CLI). A `?` will give helpful information about the commands available.

Note: The BCM shell is only used for debugging purposes and not for configuring the switch. Use the `exit` command to leave bcm mode.

Tip: It is possible to manipulate the switch settings using the bcm shell. Be cautious not to change the switch settings.

Logs

Reading logs is probably the most efficient troubleshooting method. All logs are time stamped to millisecond accuracy. All logs are available in the directory `/var/log` and alternatively can be uploaded to a log server by means of SNMP, using the GEN-LOG-MIB objects: TFTP server IP address `contTransferSrvIP`, the file path `contTransferSrvPath` and the action `transferContFile`.

Data Collection

A basic description of the problem occurring or suspected to occur in the CMX is required. Also, the following information is required for further analysis of the issue.

Steps to reproduce the issue:

1. Write down how to reproduce the situation/fault.
2. Document physical and Layer 2 and 3 logical topologies.
3. If any traffic is running, information about the flow of traffic (ingress and egress ports) is required. Traffic flow can be inspected by using the ZebOS shell commands or the bcm shell (`show counters`).
4. Ping the CMX IP address of `eth0` to `eth2` to verify contact with the board (if feasible).

All configurations done on the CMX is required. This can be obtained in the following way.

1. Save and download configuration container (using GEN-CM-MIB).



2. Perform a complete SNMP walk on the CMX using the `snmpwalk -v 2c -c NETMAN <board ip address> 1.3.6.1` command (if feasible). To run a SNMP walk from the console of the board type `snmpwalk -v 2c -c NETMAN localhost 1.3.6.1`.
3. Collect ZebOS information.

Network Commands

This section covers some basic command line (Linux shell) commands for examining basic properties of the CMX management ports and the connected network.

In order to see the network configuration you can use the `ifconfig` command:

```
root@cmxb1:~# ifconfig
```

To display the routing information, use the `route` command (below is the example with default configuration)

```
root@cmxb1:~# route -n
```

Kernel IP routing table

| Destination | Gateway | Genmask | Flags | Metric | Ref | Use | Iface |
|--------------|--------------|-----------------|-------|--------|-----|--------|-------|
| 0.0.0.0 | 192.168.14.0 | 0.0.0.0 | UG | 0 | 1 | 0 | eth0 |
| 127.0.0.0 | 0.0.0.0 | 255.0.0.0 | U | 0 | 1 | 0 | lo |
| 127.0.0.1 | 0.0.0.0 | 255.255.255.255 | UH | 0 | 1 | 410035 | lo |
| 169.254.0.0 | 0.0.0.0 | 255.255.0.0 | U | 0 | 1 | 0 | eth0 |
| 169.254.0.0 | 0.0.0.0 | 255.255.0.0 | U | 0 | 1 | 0 | eth1 |
| 192.168.14.0 | 0.0.0.0 | 255.255.255.0 | U | 0 | 1 | 2 | eth0 |

To check the properties of the CMX switch ports and the virtual networks configured on them, use the ZebOS shell and the following commands

```
cmxb1#show vlan brief
```

Bridge Group : 1

| Bridge | VLAN ID | Name | State | Member ports |
|----------|---------|---------|--------|--------------------|
| Tagged | | | | (u)-Untagged, (t)- |
| = | = | ===== | ===== | ===== |
| 1 | 1 | Default | ACTIVE | BP_1(u) BP_2(u) |
| | | | | • |
| BP_3(u) | | | | |
| BP_6(u) | | | | BP_4(u) BP_5(u) |
| BP_9(u) | | | | BP_7(u) BP_8(u) |
| BP_12(u) | | | | BP_10(u) BP_11(u) |
| BP_15(u) | | | | BP_13(u) BP_14(u) |



| | | | | |
|----------|----|----------|--------|----------------------|
| BP_18(u) | | | | BP_16(u) BP_17(u) |
| BP_21(u) | | | | BP_19(u) BP_20(u) |
| BP_24(u) | | | | BP_22(u) BP_23(u) |
| E_4(u) | | | | E_1(u) E_2(u) E_3(u) |
| E_8(u) | | | | E_5(u) E_6(u) E_7(u) |
| GE_3(u) | | | | GE_1(u) GE_2(u) |
| | | | | GE_4(u) |
| 1 | 11 | VLAN0011 | ACTIVE | BP_2(u) |

L2 Traffic Issues

There could be different kind of problems in this area. This chapter covers reduced traffic handling capabilities.

If there is any traffic loss, the Interface MIB and the GEN-PM-MIB provide the traffic counter statistics. Verify that the board receives the amount of traffic that is expected. Check also the outgoing traffic to see if the figures match. A reason for low throughput can be that there are other unexpected traffic flows that pass the switch. The traffic counters should be useful to find if any frames are discarded on any port. Are there many discarded packets? Possible reasons to discard packets include:

- Port is not a member of the virtual network.
- Packet errors.
- Port is not in xSTP forwarding state.
- Traffic overload.

Counters are also available by using the bcm shell commands such as `show counters`. The output from `show counters` can be used as a quick indication if the switch is carrying traffic or not (if they are incremented or not). A detailed study of the counters is tricky.

It can be useful to clear the counters to make it easier to follow the counter behavior. Use the following command in the bcm shell to clear the counters:

```
clear counters
```

2.2.2.2

Cloud Infrastructure

For CUDB systems deployed on a cloud infrastructure and `traceroute` stops at hop 3, then the connectivity problem is located at the cloud infrastructure. Refer to the related documentation of the infrastructure for information about network connectivity.



2.2.3 Connectivity Problem outside the CUDB System

If `tracert` stops at hop 4 or higher, then the connectivity problem is outside the CUDB system. In this case, continue troubleshooting in the IP backbone network.

2.3 Actions for HTTP Authentication Error

Check the definition of the user and password for PG endpoint (refer to [CUDB Node Configuration Data Model Description](#), Reference [3] for more information about the configuration of PG endpoints). Configure it again properly by following the steps of the [CUDB System Administrator Guide](#), Reference [2].

2.4 Common Actions after Alarm Causes are Fixed

After fixing the problems that caused automatic replay failure, do the following:

1. Perform a manual replay by following the "Replay Provisioning over CLI" procedure as described in the provisioning system library. Specific actions in the provisioning system are outside the scope of this Operating Instruction.
2. After performing the previous steps, clear the alarm manually as described in [CUDB Node Fault Management Configuration Guide](#), Reference [1].

If the troubleshooting information in this Operating Instruction is not enough to identify and solve the problem, contact the next level of support.



Glossary

For the terms, definitions, acronyms, and abbreviations used in this document, refer to [CUDB Glossary of Terms and Acronyms, Reference \[4\]](#).



Storage Engine, Provisioning Assurance Request Failed



Reference List

CUDB Documents

- [1] CUDB Node Fault Management Configuration Guide
- [2] CUDB System Administrator Guide
- [3] CUDB Node Configuration Data Model Description
- [4] CUDB Glossary of Terms and Acronyms

Other Ericsson Documents

- [5] System Safety Information
- [6] Personal Health and Safety Information